

Computer Science Department

University of Verona

A.A. 2017-18

# **Pattern Recognition**

Introduction

# Inquadramento

- Sistemi di Pattern Recognition nell'uomo:
  - riconoscere la faccia di una persona conosciuta, anche se questo cambia pettinatura, ha gli occhiali da sole, ...
  - capire quello che una persona sta dicendo, anche se varia il tono della voce;
  - leggere una lettera scritta a mano;
  - ...
- Attività che l'uomo risolve in modo molto naturale, mentre per un calcolatore sono compiti complicati e complessi

# Alcune possibili definizioni

Pattern: insieme di informazioni relative ad un'immagine o oggetto

- *Pattern recognition*
  - studio delle problematiche connesse all'utilizzo dei calcolatori per il riconoscimento automatico di dati, altrimenti detti *pattern*.
- Studio di come le macchine possono osservare l'ambiente, imparare a distinguere i pattern di interesse dall'informazione di sfondo e prendere decisioni relative alla categoria dei pattern, a partire da quella più fondamentale, ossia *classificare*
- *Sistema di Pattern Recognition*: il processo che prende in input dati grezzi (*raw*) ed effettua un'azione sulla base della “categoria” dei dati.

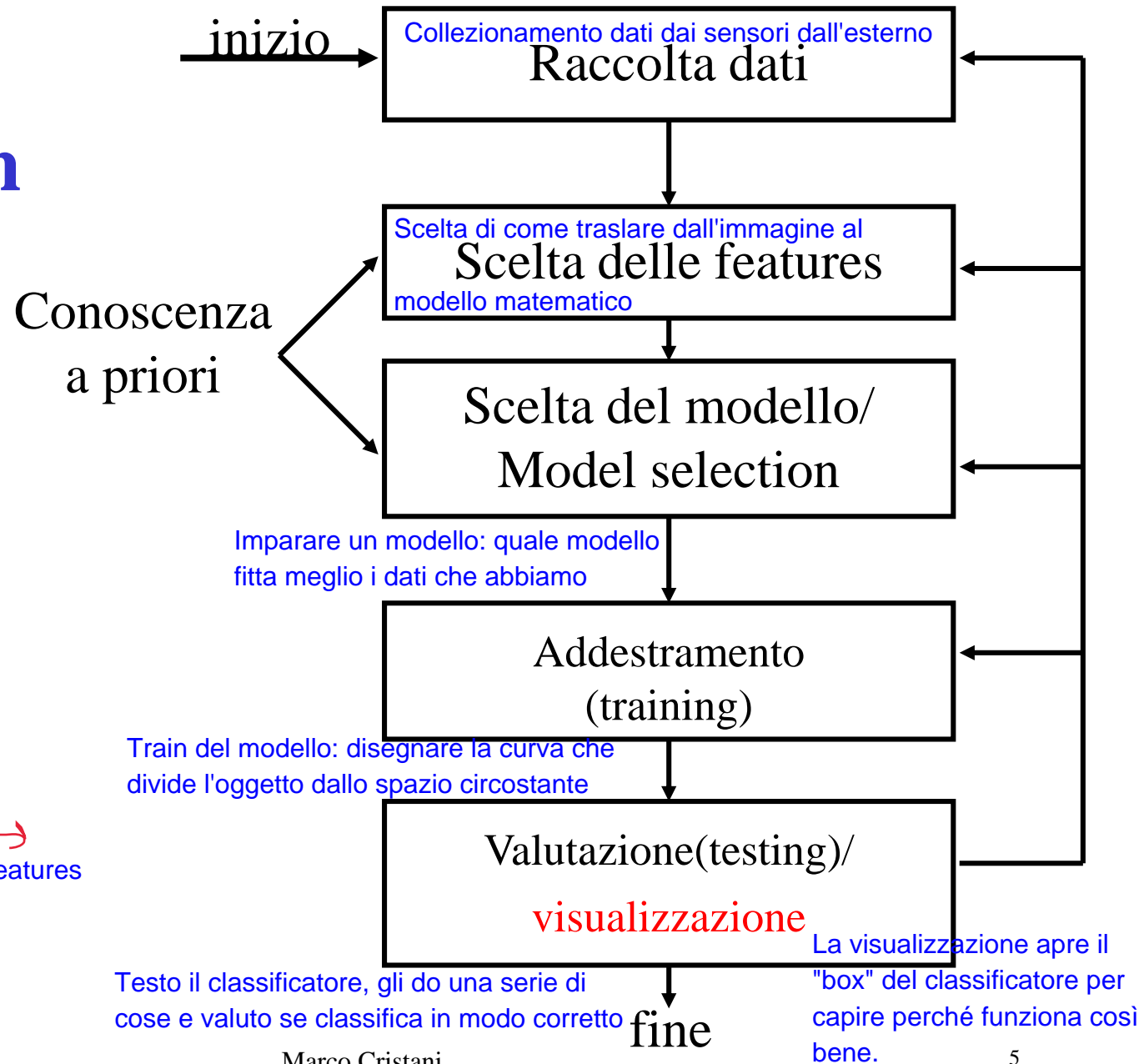
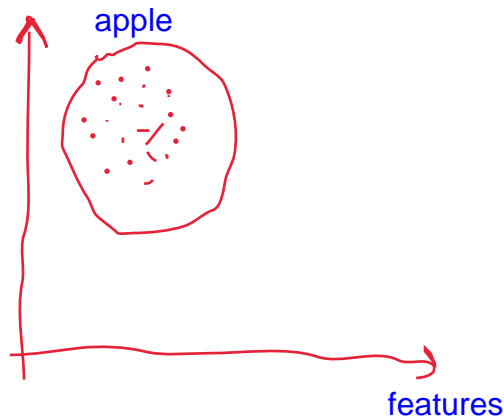
# Esempio, bilancia multimediale

Riconosce il tipo di vegetale pesato automaticamente



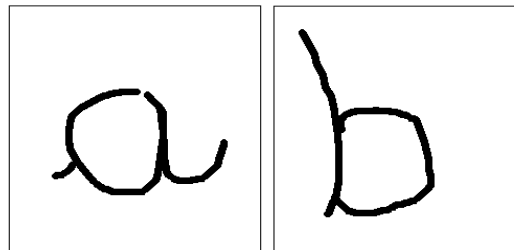
 **Fraunhofer**  
IOSB

# Sistema di Pattern Recognition



# Sistema di Pattern Recognition

*Esempio guida:* sistema che distingue tra le lettere scritte a mano “a” e “b”.



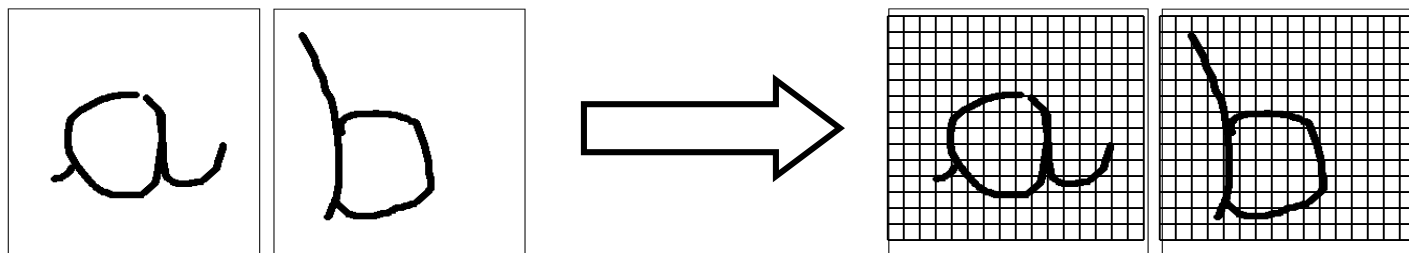
# Raccolta Dati

Per far funzionare un classificatore dobbiamo testarlo, ovvero dobbiamo raccogliere un numero sufficiente di dati di training.

- Collezione di un insieme “sufficiente” e “rappresentativo” di esempi dal problema in esame.
  - Vuol dire che deve contenere tutte le informazioni necessarie perché quel dato sia un particolare esempio di una particolare classe. Ad esempio, la classe delle lettere scritte a mano "a" non può contenere un dato che taglia parte della lettera scritta.
- “sufficiente”?
  - riesce a distinguere un pattern di test e metterlo in una delle classi a disposizione
- “rappresentativo”?
  - Vogliamo una serie di dati differenti che rappresentino adeguatamente una classe. Tornando all'esempio, deve contenere i diversi modi di scrivere "a".
  - copre lo spazio di tutti i possibili esempi
- Problemi di sensoristica (risoluzione, banda, ...)

Il modo che ho di trasmettere l'immagine nello spazio euclideo non è scontato. Ad esempio, si potrebbe trasferire l'immagine in uno spazio avente tante dimensioni quanti pixel ha l'immagine trasferita in una griglia e trovare a che punto corrisponde nello spazio multidimensionale. Però, se l'immagine successiva è leggermente spostata, i pixel a 1 saranno diversi e questo potrebbe causare grossi cambiamenti. Vedremo in seguito come rappresentare le immagini nello spazio delle features.

- *Esempio:* un insieme di immagini contenenti le lettere “a” e “b” viene acquisito tramite una telecamera, e memorizzato nel computer



L'immagine viene rappresentata da un array di pixel, ogni pixel assume valore compreso tra 0 (completamente bianco) e 1 (completamente nero)



# Scelta delle feature

- Non si possono utilizzare i dati così come sono (immagine 256x256 sono 65536 pixels)
- *Feature*: caratteristiche misurabili del fenomeno in esame (*pattern* = vettore di features):
  - semplici da calcolare;
  - invarianti a trasformazioni irrilevanti;
  - affidabili; Tutti i dati di una classe dovrebbero essere racchiusi in una parte limitata dello spazio. Devono essere significativi e non ridondanti. Ad esempio, una volta che conosco il
  - indipendenti; numero di pixel a 1 non è necessario conoscere quelli a 0. In pratica un dato è dipendente se non arricchisce le informazioni relative alla mia classe.
  - discriminanti; Devono essere in grado di distinguere adeguatamente le classi (es. numero di pixel neri non è discriminante)
  - poche (problema della *curse of dimensionality*);
- In questa fase è molto utile l'utilizzo della conoscenza a priori sul problema

## *Esempio:*

- una feature potrebbe essere il numero totale di pixel neri:
  - invariante alla rotazione e traslazione dell'oggetto
  - poco discriminante: non tiene conto della forma
- uso di meta-conoscenza a priori: devo distinguere tra “a” e “b”, e so che la lettera “b” è tipicamente più alta e allungata della “a”.
  - uso come feature il rapporto altezza/larghezza

Nella scelta delle features ci sono due famiglie di operazioni possibili da fare:

- SELEZIONE: tra i dati disponibili, devo scegliere solo quelli strettamente rappresentativi del dato.

- ESTRAZIONE: consiste nel creare un nuovo valore per una feature

## Scelta delle feature: selezione e estrazione

- Il numero di feature deve essere piccolo per limitare il costo della misura e non influire sull'accuratezza del classificatore
- Selezione di feature:
  - migliore sottoinsieme delle feature estratte
- Estrazione di feature:
  - misura sui dati
  - creazione di nuove feature da combinazioni di feature misurate (esempio precedente delle  $a$  e  $b$ )
  - Tali feature possono aver una miglior capacità discriminativa, ma si perde il significato fisico di queste.
- Uso di una funzione criterio per la riduzione: tipicamente l'errore di classificazione di un sottoinsieme di feature.
- Inoltre, è importante determinare la dimensione dello spazio ridotto.

ES: Il numero di pixel neri è un caso di selezione, ma il rapporto tra i larghezza ed altezza è un caso di estrazione.

# Scelta del modello

Posso selezionare una serie di classificatori ma devo selezionare quelli più rappresentativi del modello.

- Scelta della struttura logica e la base matematica delle regole di classificazione.
- Tipicamente, il classificatore stima, per ogni oggetto, un valore che indica il grado di appartenenza ad una o più classi sulla base del vettore di feature che lo caratterizza.
- Problemi:
  - come decidere il modello
  - come decidere la **dimensione del modello** (ossia *il numero dei suoi parametri*)
  - come capire se il modello ottenuto rappresenta effettivamente il fenomeno in esame

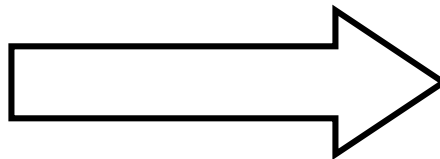
- Non esiste un classificatore che vada bene per tutte le applicazioni
- *Esempio:* uso di un classificatore a soglia:
  - data un'immagine  $I$
  - calcolo il rapporto altezza/larghezza  $E(I)$ , con  $E$  algoritmo di selezione/estrazione di feature;
  - se  $E(I)$  è maggiore di una certa soglia  $\theta$ , allora l'immagine è una “b”, altrimenti è una “a”.



# Addestramento del modello

- Sinonimi:
  - *training* del classificatore
  - *learning* del classificatore
- Processo (spesso iterativo, basato su *cicli*, *epoche*) con il quale si utilizzano i dati a disposizione (*training set*) per la costruzione del modello

Esempi tratti dal  
problema  
(*Training Set*)  
Conoscenza a  
priori



Regole che  
governano il  
fenomeno

- *Esempio:*

addestramento del modello = determinazione della soglia  $\theta$

- Si ha a disposizione una serie di immagini di esempio per le lettere “a” e per le lettere “b” (*training set*)
- calcolo  $E(I)$  per tutte le immagini del training Set
- determino una soglia  $\theta$  “adatta” a separare i valori  $E(I)$  calcolati

# Tipologie di addestramento: supervisionato

- Sinonimi: *supervised learning*, *classificazione*  
classificazione supervisionata: i dati di training sono già classificati e così
- Idea e scopo: quelli di test, in questo modo posso testare se i dati di training sono buoni  
classificatori e con quali errori vengono rilevati
  - di ogni elemento del training set si conosce l'esatta categoria.
  - L'obiettivo è quello di creare uno strumento in grado di classificare nuovi oggetti.
- Problemi:
  - capire se un algoritmo di training è capace di trovare la soluzione ottimale;
  - capire se converge, e se è sufficientemente scalabile;
  - capire se riesce a generalizzare.



- *Esempio:*
  - il training set è costituito da un insieme di immagini “a” e “b”.
  - di ogni immagine conosciamo l’esatta classificazione (cioè se è “a” oppure “b”)
  - queste informazioni sono utilizzate per determinare la soglia del classificatore.

# Tipologie di addestramento: semi supervisionato

- Sinonimi: semi-*supervised learning*,
- Idea e scopo:
  - di qualche elemento del training set si conosce l'esatta categoria, della maggior parte no
  - L'obiettivo è quello di creare uno strumento in grado di classificare nuovi oggetti.
- Problemi:
  - Come nel caso precedente

# Tipologie di addestramento: non supervisionato

Necessità di strutturare i dati senza avere una struttura.

- Sinonimi: *unsupervised learning, clustering*  
Non siamo al corrente di quante e quali classi ci siano. Ad esempio
- Idea e scopo: potremmo avere una serie di genomi e voler identificare per similitudine quelli che causano una patologia.
  - nessuna informazione sulla categorizzazione degli elementi del training set.
  - Il sistema deve trovare i clusters (gruppi) “naturali” all’interno del training set, sulla base della “similarità” tra patterns
- Problemi:
  - intrinsecamente più difficile della classificazione
  - “naturali”?
  - “similarità”?

- *Esempio:*
  - il training set è costituito da un insieme di immagini “a” e “b”.
  - nessuna informazione sulla categorizzazione delle immagini.
  - si cercano di creare due gruppi, mettendo assieme quelle immagini che hanno valore simile di  $R(I)$  (la *feature*)

# Tipologie di addestramento: Rinforzato

Faccio addestramento dei dati e poi sottometto i risultati ad un esperto che mi dice se il risultato è corretto o no.

- Sinonimi: *reinforcement learning*, *learning with a critic*
- Idea: a metà strada tra le due: non viene fornita alcuna informazione sulla categoria esatta, viene dato un giudizio sulla correttezza della classificazione
- La strategia di addestramento viene modificata:
  - si presenta un pattern al classificatore
  - il classificatore fa un tentativo di classificazione
  - viene detto *esclusivamente* se il tentativo è corretto o meno
  - sulla base del giudizio si modifica il classificatore

ES: Quando compro qualcosa Amazon mi propone ciò che potrebbe piacermi in base alle mie indicazioni. Oppure, se guardo un video su youtube e mi soffermo a guardare la pubblicità, do informazioni sul mio interesse relativo a quel marchio o a quel tipo di oggetto. Se skippo no.

# Valutazione e “visualizzazione”

- Misura delle prestazioni del classificatore
- Prestazioni di *generalizzazione*: capacità del classificatore di classificare correttamente anche esempi non presenti nel data set
- Nessun errore sul training set non implica necessariamente aver ottenuto il classificatore ottimale (pb di *overfitting*, *overtraining*)  
Abbiamo solo un certo tipo di dati che non coprono veramente tutte le casistiche.
- Per evitare situazioni di *overfitting* è sempre meglio utilizzare due insiemi disgiunti in fase di learning, uno per il training e uno per il testing.

Per evitare che il testing set contenga elementi troppo diversi dal training set, l'idea è di continuare a cambiare il training ed il testing set, scegliendo randomicamente quali elementi appartengono al primo e quali al secondo.

- Tecniche per la scelta del training set e del testing set:

Se ho  
milioni  
di dati

- *Holdout*: si suddivide casualmente il training set in due parti uguali: una per il training una per il testing
- *Averaged Holdout*: si effettuano più partizioni holdout, e si media il risultato ottenuto (*accuratezza di classificazione*). In questo modo si ha indipendenza dalla particolare partizione scelta

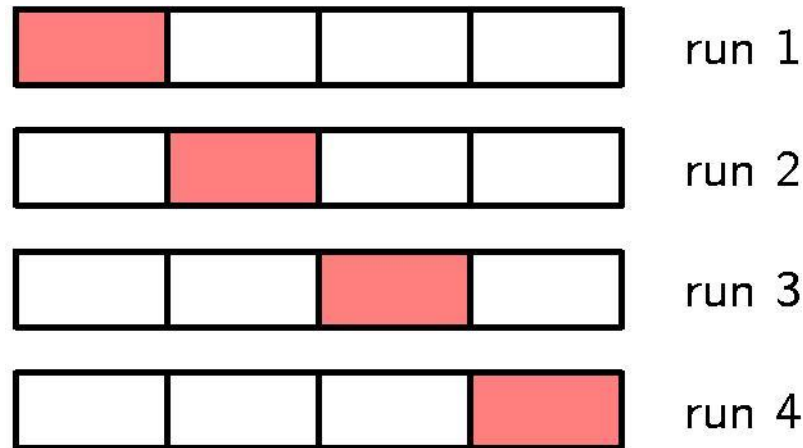
Se ho  
davvero  
pochi  
dati

- *Leave One Out*: per il training vengono utilizzati tutti i patterns tranne uno, utilizzato per il testing. Si ripete per tutte le possibili combinazioni.
- *Leave P-Out*: come il precedente, utilizza P elementi per il testing, invece che uno.

- *K-folding*: vedasi slide successiva

Partiziono in K parti e iterativamente faccio validazione prendendone uno per il testing e gli altri per il training.

- **Cross-Validation: 4-fold**

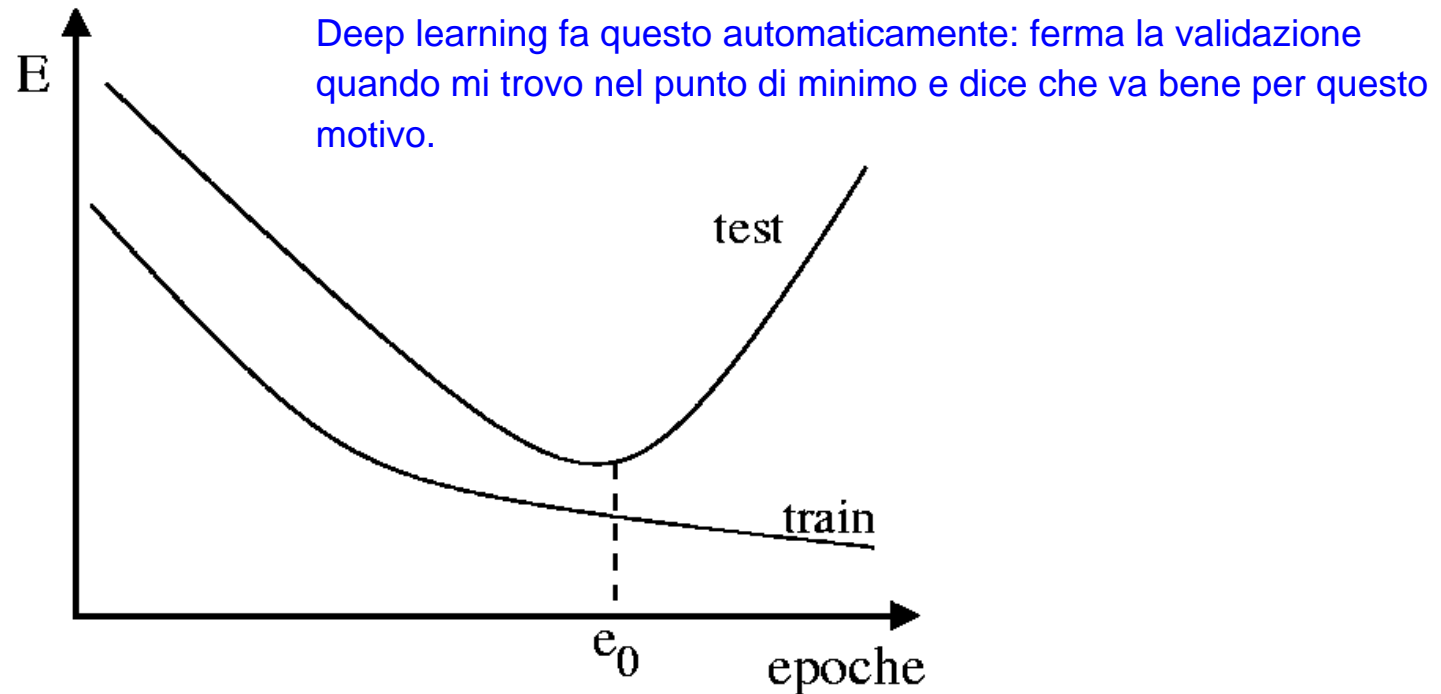


Spesso il training dataset non viene dato e si ricorre a dei "trucchetti". Ad esempio, se viene fornito solo il testing dataset, esso viene diviso e una parte viene presa come "training dataset prima del training del



# Problemi training/valutazione: overtraining

- Problema che nasce quando il classificatore impara solo i dati di training e non generalizza più



Si ferma l'addestramento prima del verificarsi del fenomeno dell'overtraining (o overfitting) ( $e_0$ )

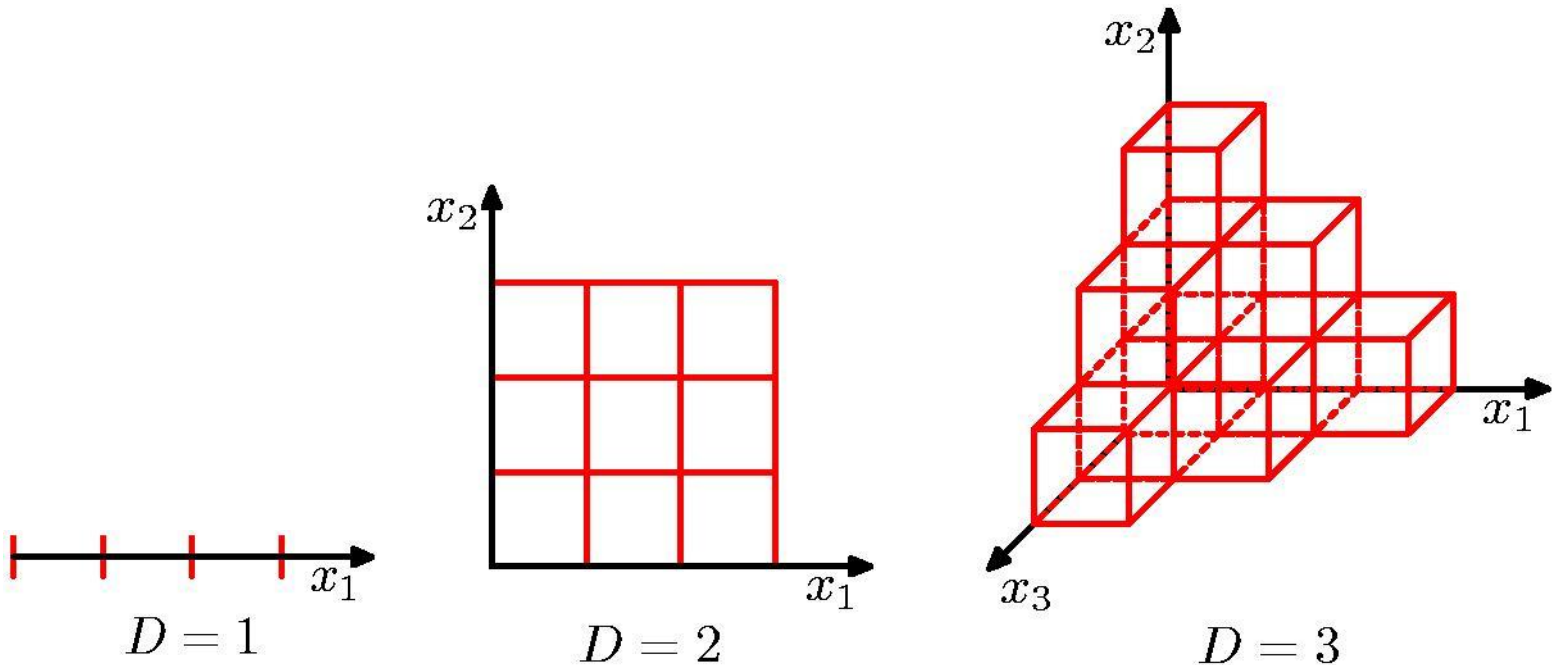
# Problemi training/valutazione: Curse of dimensionality

Quando aumento il numero di features, più sono le features meno sono le probabilità di trovare la scelta ottimale

- Le prestazioni dipendono dalle relazioni tra il numero di campioni, numero di feature e dalla complessità del classificatore.
- In teoria, la probabilità di errore non aumenta se si aggiungono feature
- In pratica si riscontrano dei problemi dovuti al fatto che le ipotesi sono solo approssimazioni nei casi reali
- Inoltre, il numero di campioni deve essere in relazione esponenziale rispetto al numero di feature
- Tutti i comuni classificatori soffrono di questo problema ed esistono regole guida

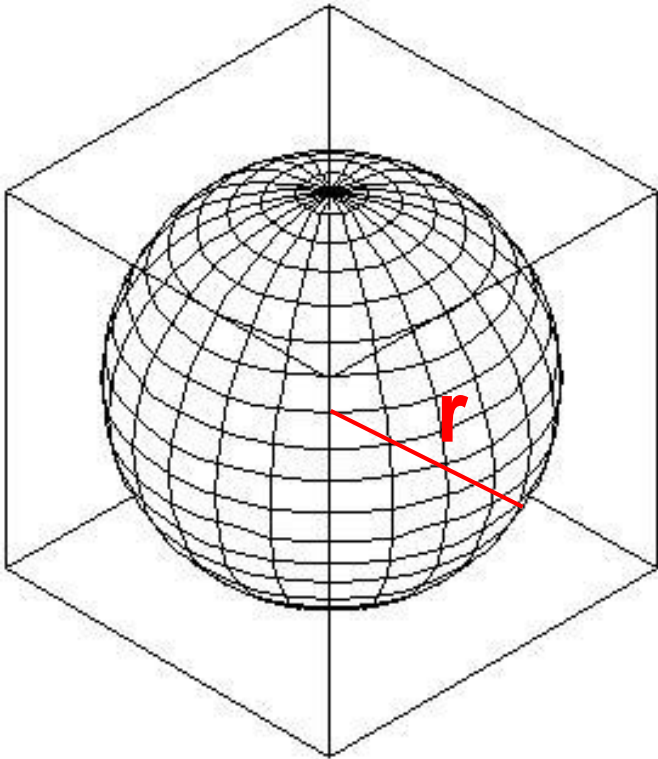
# Problemi training/valutazione: Curse of dimensionality (2)

- Vorrei tessellare lo spazio di esistenza dei dati in modo uniforme. Di quanti dati ho bisogno, all'aumentare delle dimensioni?



# Problemi training/valutazione: Curse of dimensionality (3)

Mostro che avere i dati nello spazio sferico nel cubo che lo racchiude è non informativo ad alte dimensionalità



$$\text{Vol. cubo} = r^d$$

$$\text{Vol. sfera} = \frac{r^d \pi^{d/2}}{\Gamma(d/2 + 1)}$$

Metto a rapporto e al limite ottengo  $f_d = \lim_{d \rightarrow \infty} \frac{\pi^{d/2}}{d 2^d \Gamma(d/2 + 1)} = 0$

# Problemi training/valutazione: Curse of dimensionality (4)

d	1	2	3	4	5	6	7
$f_d$	1	.785	.524	.308	.164	.08	.037

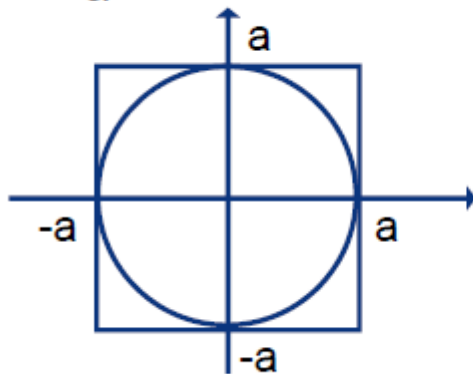
Mano a mano che la dimensionalità cresce, il volume della sfera è molto più piccolo di quello del cubo

1.  $d = 1$



volume is the same

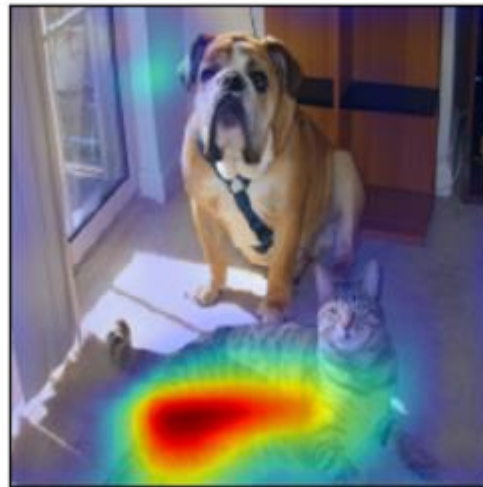
2.  $d = 2$



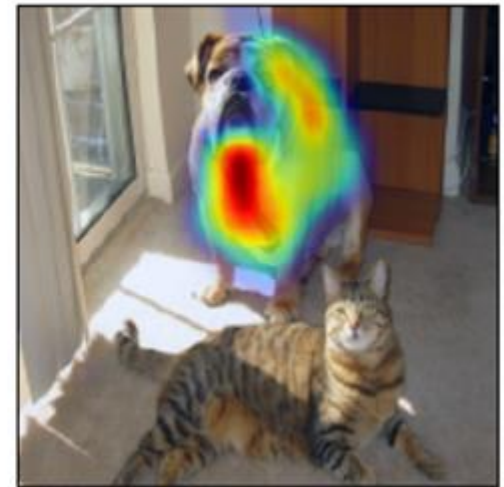
volume of sphere is already smaller

# Visualizzazione

- Operazioni di visualizzazione permettono di capire in maniera interpretabile dall'uomo perché un classificatore ha performato bene/male
- Sono nati nel 2016 espressamente per capire come operano i classificatori basati su deep learning



cat

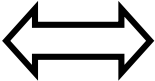
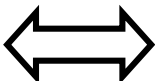


dog

# Approcci alla Pattern Recognition

- **Approccio sintattico:** approccio gerarchico.

Analogia tra la struttura dei patterns e la sintassi di un linguaggio:

– patterns		frasi di un linguaggio
– sottopattern primitivi		alfabeto

- **Approccio statistico:**

- ad ogni pattern viene associato un vettore di feature che rappresenta un punto nello spazio multidimensionale del problema.
- L'informazione sul problema, le dipendenze tra i vari fattori e i risultati prodotti sono tutti espressi in termini di probabilità.

# Classificazione statistica

- La descrizione statistica di oggetti utilizza descrizioni numeriche elementari chiamate *feature*, che formano i cosiddetti *pattern*,  $\mathbf{x} = (x_1, x_2, \dots, x_n)$  o vettori di *feature*.
- L'insieme di tutti i possibili pattern forma lo spazio dei pattern o delle *feature*.
- Nella classificazione statistica, alle entità di interesse sono associate delle distribuzioni di probabilità
- Un pattern è un'entità di interesse, quindi anche ad esso è associata una probabilità!



# Classificatore di Bayes

- Framework di base per la classificazione statistica
- Lavagna

$$P(w|x) = \frac{P(x|w)P(w)}{P(x)=e}$$

e=evidence  
w:=class  
x:=data instance,  
observation

Posterior Probability=(Likelihood probability)(Prior probability)/  
evidence

Il risultato statistico  
del classificatore sta  
nell'intervallo [0,1]

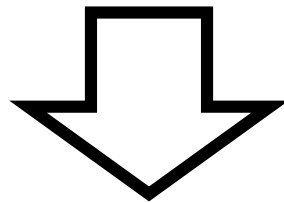
$$P(x) = \sum_i^C P(x|w_i)P(w_i)$$

# Stima delle pdf

- *Classificatori parametrici*
  - si fissa il modello della distribuzione e sulla base del training set se ne stimano i parametri;
  - esempio: classificatore gaussiano.
- *Classificatori non parametrici*
  - nessuna assunzione sulla forma della pdf, la stima si basa esclusivamente sui dati;
  - esempio: K-nearest Neighbor.
- *Classificatori semi-parametrici*
  - si ha una classe molto generale di modelli di pdf, in cui il numero di parametri può essere aumentato in modo sistematico per costruire modelli sempre più flessibili;
  - esempio: reti neurali.

# K Nearest Neighbor (KNN)

- Classificatore non parametrico.
- Molto utilizzato per la sua semplicità, flessibilità e ragionevole accuratezza dei risultati prodotti.
- IDEA: due elementi della stessa classe avranno, molto probabilmente, caratteristiche simili, cioè saranno vicini nello spazio dei punti che rappresenta il problema



La classe di un punto può essere determinata analizzando la classe dei punti in un suo intorno

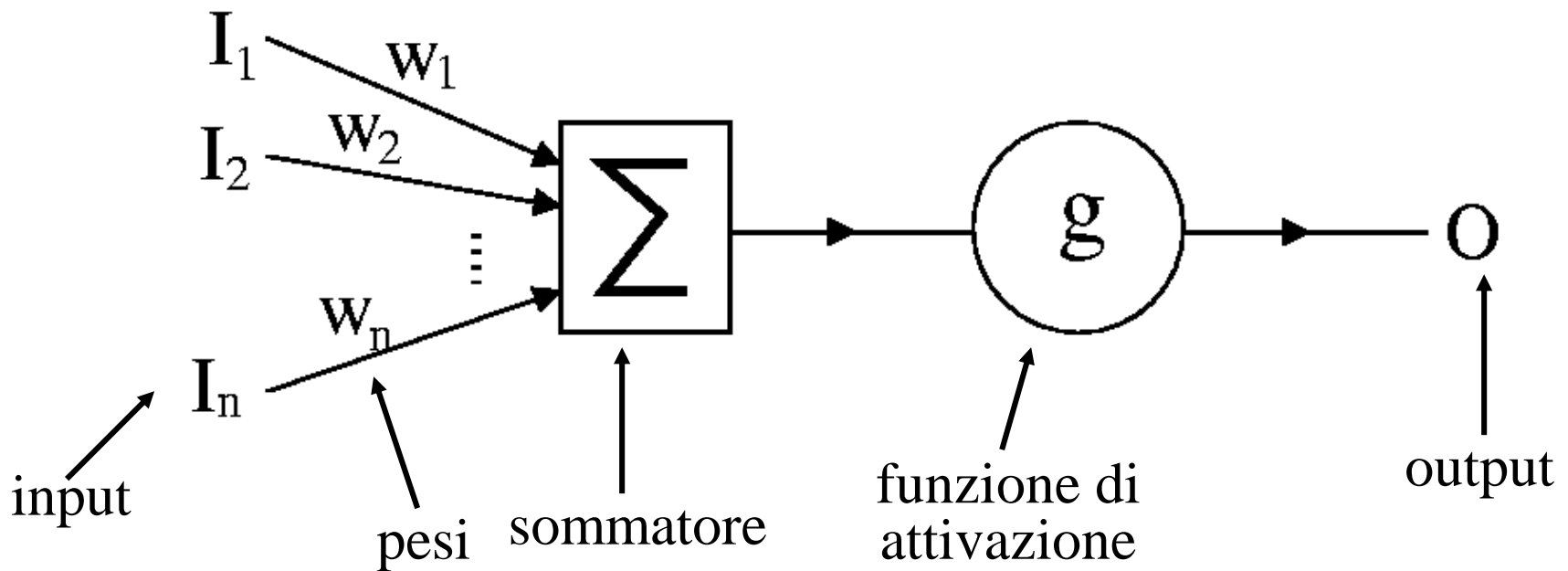
# Algoritmo

- Dato un insieme di esempi  $X$ , dato un punto da classificare  $x_0$ :
  - si calcola l'insieme  $U$  dei  $K$  punti di  $X$  più vicini a  $x_0$  secondo una determinata metrica  $\Sigma$  (di solito la distanza euclidea);
  - si calcola la classe  $C$  più frequente all'interno dell'insieme  $U$ ;
  - $x_0$  verrà classificato come appartenente a  $C$ .
- Problema: scelta del parametro  $K$  e della metrica  $\Sigma$ .

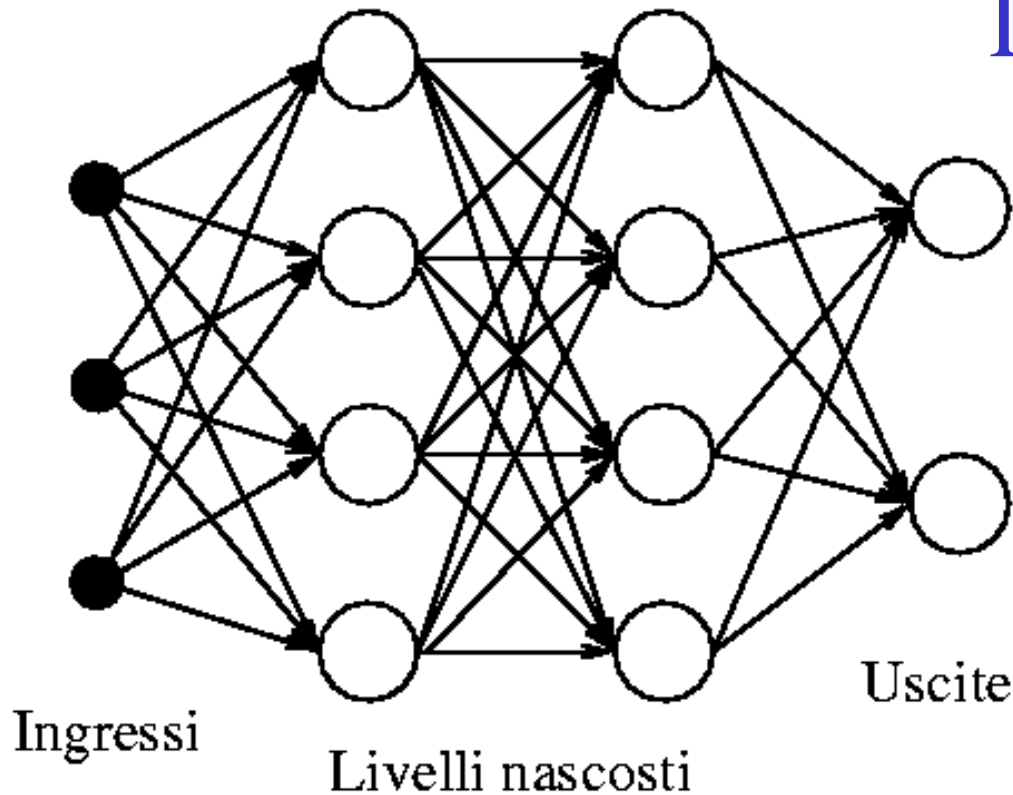
# Reti neurali, un caso (quasi) a parte

- Sistema artificiale di elaborazione dell'informazione che emula il sistema nervoso animale.
- Caratteristiche del sistema nervoso animale:
  - robusto e resistente ai guasti;
  - flessibile, si adatta a situazioni nuove imparando;
  - lavora anche con informazione approssimata, incompleta o affetta da errore;
  - permette un calcolo altamente parallelo;
  - piccolo e compatto.

- Reti neurali: struttura complessa, composta da tante unità elementari di calcolo collegate tra loro in vario modo.
- Le unità elementari sono dette *neuroni*.
- I collegamenti sono detti *sinapsi*.



## Diverse topologie



*Feed forward  
neural networks*

# Combinazioni di classificatori

- Problema poco studiato e solo recentemente ripreso.
- Assumono
  - classificatori diversi e con prestazioni diverse e non ottime;
  - training set diversi;
  - diversi classificatori addestrati con uguali training set e quindi con prestazioni diverse;
  - ugual classificatori addestrati differentemente (NN) e che risultano avere differenti prestazioni.
- Obiettivo di aumentare le prestazioni
- Approcci:
  - parallelo: si combinano i risultati dei singoli classificatori;
  - seriale: i risultati di uno sono input del successivo fino al risultato finale;
  - gerarchico: i classificatori sono strutturati ad albero e i risultati combinati adeguatamente.



# In sintesi ...

- ... nell'uso di questi algoritmi di apprendimento bisogna tener conto di
  - complessità del modello (*model selection*)
  - fase di addestramento *online* o *batch*
  - disponibilità e tipo di dati di training, test e validazione
  - meccanismi di stop della fase di learning (pb di *overfitting*)
  - valutare l'uso di diversi modelli o dello stesso tipo di modello addestrato in modo diverso (insiemi di classificatori)
  - bilanciamento dei dati a disposizione
- Spesso, tutti questi modelli sono **adattati** allo specifico problema/applicazione da affrontare, sono spesso necessarie **metodi e modelli *ad hoc***

# APPLICAZIONI

# The Pattern Recognition Era

- Fattori che hanno decretato il decollo definitivo della pattern recognition negli ultimi anni:
  - aumento della capacità computazionale dei calcolatori, deep learning.
  - presenza di grosse quantità di dati
    - Internet
    - Social media (Facebook, Flickr, Pinterest)
  - nuovi sistemi di interazione uomo-macchina, uomo-robot, domotica.

Today...

*SceneUnderLight*



**OSRAM**



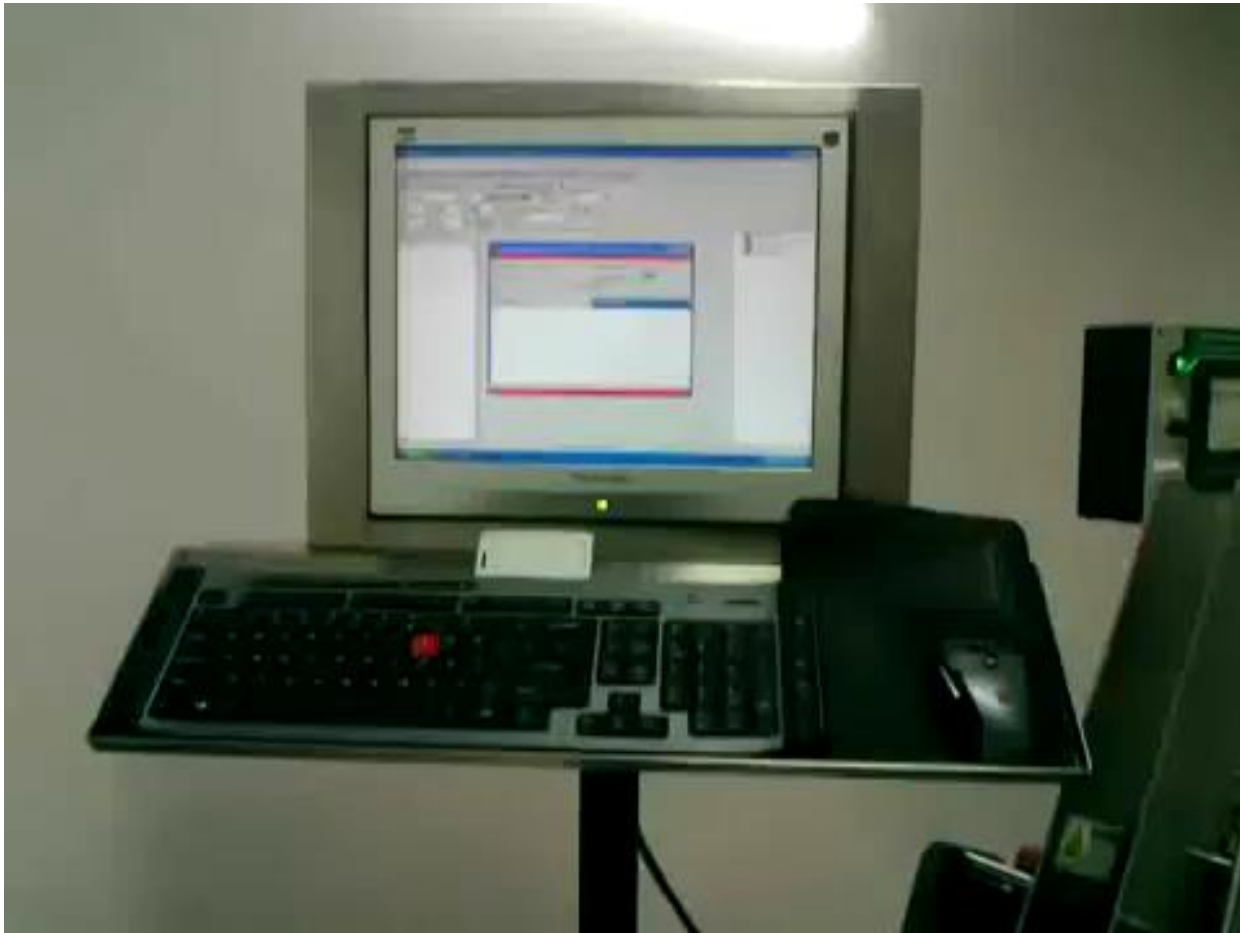
ISTITUTO  
ITALIANO DI  
TECNOLOGIA

Supplementary material for  
submission 3292

***MX-LSTM:***

***mixing tracklets and vislets to jointly  
forecast trajectories and head poses***

# Controllo qualità



# Autonomous driving



# Autonomous driving – pedestrian detection





# Robust classification

<https://www.theedgecompany.net/>



# Active Object Recognition

